User's Guide

TBPW 1.1

Dyutiman Das, William Mattson, Nichols A. Romero, Richard M. Martin

University of Illinois at Urbana-Champaign
Materials Computation Center

July 7, 2004

Available on-line at
http://www.mcc.uiuc.edu/software/
http://ElectronicStructure.org

Coordinated with the book:
"Electronic Structure: Basic Theory and Practical Methods," by Richard M. Martin,
Cambridge University Press, 2004.
http://books.cambridge.org/0521782856.htm

Comments
tbpw@mcc.uiuc.edu

1 INTRODUCTION

TBPW is an electronic structure code primarily intended for pedagogical purposes. It is written from the ground-up in a modular style using Fortran 90. This code is composed of three distinct parts: empirical tight binding (**TB**), empirical pseudopotential plane wave (**PW**), and features common to all band structure codes (**Common**).

The main characteristics of these codes are:

- Readily provides band structure plots
- **TB** implements the Slater-Koster 2-center formalism. Options are provided for simple models (Harrison or user specified) or a completely general approach using a rotation matrix formalism that allows the use of orbitals with arbitrary angular momentum (ℓ).
- **PW** implements the plane wave method for any local potential or pseudopotential. Several choices for potentials are provided and the user can add others. The bands can be found using direct diagonalization or a conjugate gradient method. Additionally, there is a plane wave density (**PWD**) code which outputs the electron density on a grid.

The code is coordinated with the description given in the book and associated web site:

- **Electronic Structure: Basic Theory and Practical Methods**, by Richard M. Martin, Cambridge University Press, 2004.
  (See http://books.cambridge.org/0521782856.htm).
- **ElectronicStructure.org** is the web site (http://ElectronicStructure.org) for new material and updates, including links to **TBPW** and many other codes.

The input uses keywords ("tags") and input data chosen to coordinate with those used by the **SIESTA** code (http://www.uam.es/departamentos/ciencias/fismateriac/siesta/), which is a full self-consistent *ab initio* pseudopotential local orbital code.

Two technical papers describing the methods implemented in this code are located in `Doc`.

- `tight_binding.pdf` describes tight binding plus the rotation matrix implementation used to handle orbitals of arbitrary angular momentum.
- `conj_grad.pdf` describes the conjugate gradient method used in the PW code.

2 QUICK START

2.1 Compilation

There is a machine independent makefile included in the source root directory. `Make` should always be run from this directory. The machine dependent makefile are located in directory named `Sys`. Find a suitable makefile for your architecture, and then copy or link to `arch.make` located in the source root directory. Note that `arch.make` is included by the main makefile.

The sources files are organized in three directories: `TB`, `PW`, and `Common`. Modules which are shared by both **TB** and **PW** codes are placed in the `Common` directory.

There are three targets to make:

- **tb** – the tight binding code
- **pw** – the plane wave code
- **pwDensity** – the plane wave code that produces the electron density on a grid

Typing `make` plus the target name will create the create the executable in its respective source directory.

2.2 Libraries

TBPW is linked against the LAPACK and BLAS libraries. They can be downloaded from `http://www.netlib.org/`. The appropriate variables in the machine dependent makefile will require modification. Because different compilers handle calls to external functions differently, the following two preprocessor flags are available for adding underscores to LAPACK (-DAUSLAPCK) and BLAS (-DAUSBLAS). If you get undefined reference errors in the external subroutine calls to LAPACK or BLAS, try using these flags.

2.3 Running the program

Some example input files along with their corresponding bands structure plots are provided in the directory `TBPW/TB/Examples` and `TBPW/PW/Examples`. The tight binding and plane wave codes are executed follows:

```
./pw
```

`OR`

```
./tb
```

At the `Input File Name` prompt, type the filename including any extension. The band structure output file can be easily plotted with gnuplot by typing

```
gnuplot gnuplot.dat
```

This also creates a band structure plot file `band.ps`

2.4 Getting help with the code

To post questions or comments about **TBPW** send an e-mail to tbpw@mcc.uiuc.edu. You can subscribe to this mailing list by sending e-mail to tbpw-subscribe@mcc.uiuc.edu.

3 REVISION HISTORY

3.1 Changes with respect to version 1.0

1. The tag **DiagonalizationSwitch** no longer supports option 2 - Conjugate gradient method + FFT. As a result, **TBPW** does not need the FFTW library.

2. The value for the tag **LminAndLmax** does not need to be specified in the input file explictly. It is determined by the program at run time.

3. Tag **Channel** renamed to **NumChannelsPerSpecie**.

4. Introduction of new tag **TightBindingModelType**.

4 INPUT DATA FILE

4.1 Format of Input File

All information is read in using the "Taghandler" routine to search for a "tag" and then reading the desired information on the same line as the tag or the following line(s). Tags are listed below with description of the data to be read in, example inputs, and default value(s). Tags can be in any order and the Taghandler" routine always finds only the first occurrence of a tag. Tags listed under section **Tight binding**, **Plane wave**, and **Plane wave density** are specific to their respective methods. TB specific tags are ignored by PW and PWD, PW specific tags are ignored by PWD and TB, etc.

The tags are chosen to coordinate with those used by the SIESTA code, http://www.uam.es/departamentos/ciencias/fismateriac/siesta/.

Except were noted otherwise, all input and output parameters are in atomic units: energy in Hartrees and length in Bohr.

4.2 General System Descriptors

NumberOfAtoms (*integer*): Number of atoms per primitive cell in the calculation. For an isolated system this is the total number of atoms.

```
NumberOfAtoms 2
```

Default value: No default. Variable must be supplied.

NumberOfDimensions (*integer*): This defines the dimension of the space used the calculation. Lattice vectors and basis vectors (atom coordinates) are stored in arrays of dimension ndim read in after NumberOfDimensions tag. In PW this means the space dimension, which can be any dimension $1, 2, 3, \ldots$. In TB this defines the dimension of the lattive and atomic coordinates; the orbitals included in the calculation are defined by the model and/or input file. (For example, a line of atoms is defined by ndim =1, whereas a nanotube is defined in a 3-dimensional space with ndim =1. For a nanotube, the orbitals can be chosen to be the full set of s, p_x, p_y, p_z, or one can treat only the π orbitals, etc.)

```
NumberOfDimensions 3
```

Default value: 3

NumberOfSpecies (*integer*): Number of different atomic species in the calculation. Atoms of the same species, but with different potential parameters are counted as different species. As of this moment, multiple species have no function in the PW code.

```
NumberOfSpecies 2
```

Default value: 1

ChemicalSpeciesLabel (*data block*): It specifies the different chemical species that are present, assigning them a number for further identification.

```
ChemicalSpeciesLabel
1 31 Ga
2 33 As
```

The first number in the line is the species number (assumed to be in sequential order), it is followed by the atomic number, and then by the desired label.

In PW the label must be one of the labels for which there is a defined potential. At present, potentials provided in the code are limited to: Ga, As, and Si (empirical pseudopotentials); H (Coulomb potential screened with Thomas-Fermi function for $r_s = 1.0$); and `El` (Empty lattice - free electrons with zero potential). Additional potentials can be added to module located in `atomPotentialMod.f90`.

In TB the label merely is a convenient label for the species. For example one can chose input such as

```
ChemicalSpeciesLabel
 1  14   Si
 2  14   Si_surface
```

Default value: No default. Input must be supplied.

4.3 Lattice and coordinates

LatticeConstant (*real*): This defines the scale of the lattice vectors (Bohr).

```
LatticeConstant 10.6769569
```

Default value: 1 Bohr

LatticeVectors (*data block*): The lattices vectors are read in units of the lattice constant defined above. There are NumberOfDimensions basis vectors each with NumberOfDimensions components.

```
LatticeVectors
0.0 0.5 0.5
0.5 0.0 0.5
0.5 0.5 0.0
```

Default value: No default. Variable must be supplied.

AtomicCoordinatesFormat (*string*): Character string to specify the format of the atomic positions in input. These can be expressed in two forms:

- `ScaledCartesian` (atomic positions are given in Cartesian coordinates, in units of the lattice constant)
- `ScaledByLatticeVectors` (atomic positions are given referred to the lattice vectors)

Default value: `ScaledByLatticeVectors`

AtomicCoordinatesAndAtomicSpecies (*data block*): The data is read as follows:

```
From i = 1 to totalNumAtoms
     read: atomCoordinates(:,i) speciesOfAtom(i)
```

Default value: No default. Variable must be supplied.

InputEnergiesInEV : This tag does not store any value. If this tag is found, the input energies are in eV. If this tag is omitted, input energies are in Hartrees. This effects **OrbitsAndEnergies**, **SKMatrices** in TB and **EnergyCutoff** in PW. Output energies are still in Hartrees.

4.4 k-point sampling and band structure plots

KPointsScale (*string*): Specifies the scale of the k-vectors given in the **KPointsAndLabels** below. The available options are:

- `TwoPi/a` (k-point vector coordinates are given in Cartesian coordinates, in units of $2\pi/a$, where a is the lattice constant)
- `ReciprocalLatticeVectors` (k-point vectors are given in reciprocal lattice vector coordinates)

Default value: `ReciprocalLatticeVectors`

NumberOfKPoints (*data block*): Read in **NumberOfKPoints** from the input file in units determined by **KPointsScale**.

```
NumberOfKPoints
2
0.0  0.0  0.0
0.25 0.25 0.25
```

Note that this tag is incompatible with **KPointsAndLabels**. For the moment, this option has no function outside **PWD** code.

Default value: No default. Variable must be supplied.

KPointsAndLabels (*data block*): Specifies the lines along which band energies are calculated. Here is an example for FCC lattice using the option `ReciprocalLatticeVectors`:

```
KPointsAndLabels
0.0   0.0   0.0      Ga
0.375 0.375 0.75     K
0.5   0.5   0.5      L
0.0   0.0   0.0      Ga
0.0   0.5   0.5      X
0.25  0.625 0.625    U
```

The last column (mandatory) is a LaTeX label for use in the band plot. Note that this tag is incompatible with **NumberOfKPoints**. This option has no useful function outside of **PW** and **TB** code.

Default value: No default. Variable must be supplied.

NumberOfLines (*integer*): Number of directions in the reciprocal lattice along which the bands are to be plotted. This value cannot exceed the number of special k-points used **KPointsAndLabels** minus one. For example, if **NumberOfLines** equals 4, the band structure is only calculated from Ga – X in the aforementioned FCC lattice. This tag is required if using **KPointsAndLabels**.

```
NumberOfLines 5
```

Default value: No default. Variable must be supplied.

NumberOfDivisions (*integer*): The number of divisions per line in k-space. This tag is required if using **KPointsAndLabels**.

```
NumberOfDivisions 15
```

NumberOfBands 10 *Default value:* No default. Variable must be supplied.

NumberOfBands (*integer*): The number of bands to plot must be less than the total number of orbitals (*i.e.* basis).

```
NumberOfBands 10
```

Default value: No default. Variable must be supplied.

4.5 Tight binding

TightBindingModelType (*integer*): Choice of model:

- `0` General Method using rotation matrices that works for any angular momentum. Requires input number of channels for each specie and input of Slater-Koster arrays with F and K parameters. See below for description.

- `1` Harrison universal parameters. See Harrison, "Elec. Str. and Props. of Solids". Assumes only one channel for each specie (default value - no need for input of number of channels. Must read in on-site energies using OrbitsAndEnergies. No other parameters are read in; two center integrals are defined solely in terms of universal parameters and the distances between neighbors.
- `10 or greater` Special models defined by the subroutine `TBParamsSpecial`. Parameters are read in using the TightBindingParameters tag. This is the only routine that must be changed to define a new model. Examples include:
  - `10` Any crystal with one s-state per site and only one nearest-neighbor 2-center hopping matrix element. The value is defined as first entry under TightBindingParameters which applies to all neighbors below a cutoff radius defined by the second entry under TightBindingParameters. All other information is in the structure and the OrbitsAndEnergies. Examples that can be treated this way include any of the s-bands for elemental crystals in texts such as Ashcroft and Mermin, Kittel, etc., and π bands of a graphene layer and a buckyball or nanotube in the pure π state approximation.

    ```
    TightBindingParameters
    2
    1.0
    1.1
    ```
  - `11` Same as above with first and second neighbor 2-center hopping matrix elements. The values are defined under TightBindingParameters: Entry 1 - n.n. matrix element; Entry 2 - cutoff distance for n.n.; Entry 3 - n.n.n. matrix element; Entry 4 - cutoff distance for n.n.n.

    ```
    TightBindingParameters
    4
    1.0
    1.1
    0.3
    2.2
    ```
  - `12` A model of an ionic crystal with two atoms per cell, one with one s-state and one with p-states. The value is defined under TightBindingParameters: Entry 1 - n.n. s-p sigma matrix element; Entry 2 - cutoff distance for n.n. The onsite energies are defined under OrbitsAndEnergies.

```
TightBindingModelType  1
```

Default value: 0 (General Method - requires input of Slater-Koster arrays with F and K parameters)

MaximumDistance (*real*): The interaction cutoff distance in Bohr units.

```
MaximumDistance 5.5
```

Default value: No default. Variable must be supplied.

NumChannelsPerSpecie (*data block*): The total number of electron channels for each chemical species. The distinct channels corresponds to the different principal quantum numbers.

```
NumChannelsPerSpecie
1 2
```

The first number is the **ChemicalSpeciesLabel** while the second number is the total number of channels. If the chemical species were nitrogen for example, the first channel would be the 1s e^- while the second channel would include the 2s and 2p e^-. *Note that the number of channels does not impose any restriction on the number of available orbitals.*

Default value: Default = 1.

OrbitsAndEnergies (*data block*): The on-site Slater-Koster parameters (also called "orbital energies") for each species, channel and specific orbital are specified in units determined by the switch **InputEnergiesInEV**.

```
OrbitsAndEnergies
1 1 4
0  0  -4.2
1  1   1.715
1  2   1.715
1  3   1.715
1 2 1
0  0   6.68
```

In the example above, the first and sixth line provides the *species number*, *channel number*, and *number of orbitals*. The *number of orbitals* does refers to the number orbitals in the basis for that particular specie and channel. It does not refer to the number of occupied orbitals.

The second through fifth line and the seventh line specify the *angular momentum*, *orbital ID*, and *orbital energy*. There will be such a line for each orbital.

The order of the orbitals is defined for each model type:

`0` For type 0 (General rotation matrix method) the orbitals are real spherical harmonics. For each angular momentum ℓ, $2\ell + 1$ orbitals must be listed. The order is given below.

`1` For type 1 (Harrison universal model) the orbitals are real spherical harmonics are the same as for model 0, except that only up to $\ell = 1$ is supported at present.

`>=10` These are specially coded models and it is up to the user to determine the order and the meaning of the orbitals.

For a given ℓ, all $2\ell+1$ orbital energies must be specified. The only allowed *orbital ID* for $\ell = 0$ is 0, for $\ell = 1$ is 1 thru 3, for $\ell = 2$ is 4 thru 8, for ℓ is ℓ^2 thru $\ell^2 + 2\ell$. The orbital ID refers to the real orbitals constructed from linear combinations of the spherical harmonics (Y_{lm}). The precise order of the orbitals is defined as done by many authors, e.g., see

`http://www.cachesoftware.com/mopac/Mopac2002manual/real_spherical_harmonics.html`

for the expressions for the orbitals in order for ℓ up to 3. However, other authors may use a different order. For a particular ℓ, the orbital ID ℓ^2 refer to the real orbital of the form $Y_{\ell 0}$. The next orbital ID, $\ell^2 + 1$, refers to the real orbital of the form $(Y_{\ell-1} - Y^*_{\ell+1})/\sqrt{2}$. The next next orbital ID, $\ell^2 + 2$, refers to the real orbital of the form $i(Y_{\ell-1} + Y^*_{\ell+1})/\sqrt{2}$. After the real orbital whose ID equals ℓ^2, the orbitals alternate between those of the form

$$\frac{1}{\sqrt{2}}\left(Y_{l-m} + (-1)^m Y^*_{lm}\right)$$

and those of the form

$$\frac{i}{\sqrt{2}}\left(Y_{l-m} - (-1)^m Y^*_{lm}\right)$$

where m is the azimuthal quantum number.

Table 1: Orbital ID for $\ell = 0$ thru $\ell = 2$

orbital ID	real orbital
0	s
1	p_z
2	p_x
3	p_y
4	$d_{3z^2-r^2}$
5	d_{xz}
6	d_{yz}
7	$d_{x^2-y^2}$
8	d_{xy}

Default value: No default. Variable must be supplied.

numSKparams (*integer*): (Used only for TightBindingModelType 0 - General rotation method.) Number of Slater-Koster parameters to be read in. It is only necessary to specify the non-zero Slater-Koster values.

```
numSKparams 5
```

Default value: No default. Variable must be supplied for TightBindingModelType 0.

SKMatrices (*data block*): (Used only for TightBindingModelType 0 - General rotation method.) There will be **numSKparams** lines with the following input:
ℓ_i ℓ_j m n_i n_j species$_i$ species$_j$ F K

```
SKMatrices
0 0 0 1 1 1 1  0.0 -2.075
0 1 0 1 1 1 1  0.0  2.480816372
1 1 0 1 1 1 1  0.0  2.71625
1 1 1 1 1 1 1  0.0 -0.715
1 0 0 1 2 1 1  0.0 -2.327399971
```

F contains the distance dependent part of the Slater-Koster matrix, while K contains the distance independent part. The Slater-Koster matrix is

$$K \times R_{ij}^F$$

where R_{ij} is the separation between atoms i and j. where F is a real number and K is units of determined by **InputEnergiesInEV** times Bohr^{-F}.

The Slater-Koster matrix has many symmetries. It is diagonal in m and since $\pm m$ are equivalent only $|m|$ is needed. The following symmetries are also imposed,

$$\begin{aligned} F(\ell_j, \ell_i, m, n_j, n_i, \text{species}_j, \text{species}_i) &= F(\ell_i, \ell_j, m, n_i, n_j, \text{species}_i, \text{species}_j) \\ K(\ell_j, \ell_i, m, n_j, n_i, \text{species}_j, \text{species}_i) &= (-1)^{\ell_i+\ell_j} * K(\ell_i, \ell_j, m, n_i, n_j, \text{species}_i, \text{species}_j) \end{aligned}$$

Default value: Any Slater-Koster parameter not specified is set to zero. Used only for TightBindingModelType 0.

4.6 Plane wave

PW carries out calculations of bands using a plane wave basis. The calculation is done with a fixed potential (not self-consistent) which may be any form of a spherical potential $V(r)$ around each atom site. For realistic calculations the potential must be chosen to be a reasonable representation of the true total potential or pseudopotential in the crystal. The potential is specified by the label **ChemicalSpeciesLabel** in which must be one of the labels for which there is a defined potential. At present, potentials provided in the code are limited to: Ga, As, and Si (empirical pseudopotentials); H (Coulomb potential screened with Thomas-Fermi function for $r_s = 1.0$); and `El` (Empty lattice - free electrons with zero potential). Additional potentials can be added to module located in `atomPotentialMod.f90`.

EnergyCutoff (*real*): Defines the plane wave cutoff (in Hartree units unless the tag **InputEnergyInEV** is present in the input file).

```
EnergyCutOff 6.0
```

Default value: No default. Variable must be supplied.

DiagonalizationSwitch (*integer*): The available methods for calculating eigenvalues and eigenvectors are:

`0` – Direct matrix inversion

`1` – Conjugate gradient method

Conjugate gradient is more efficient than direct matrix inversion for matrices of rank 700 or larger.

```
DiagonalizationSwitch 0
```

Default value: 0

CGIterationPeriod (*integer*): Occasionally, the initial trial eigenvector is "bad." **CGIterationPeriod** is the number of retries. Typical values for this tag are between 1 and 5.

```
CGIterationPeriod 1
```

Default value: 1

CGTolerance (*real*): Specifies the tolerance on the energy eigenvalues; in Hartree unless the tag **InputEnergyInEV** is present in the input file.

```
CGTolerance 1.d-4
```

Default value: No default. Variable must be supplied.

4.7 Plane wave density

MonkHorstPack (*data block*): The number of division per dimension to sample in the first Brillouin zone.

```
MonkHorstPack
2 2 2
```

Default value: Gamma point of the cell.

NumberOfOccupiedBands (*integer*): 2×**NumberOfOccupiedBands** must equal the number of electrons.

Default value: No default. Variable must be supplied.

5 ACKNOWLEDGEMENTS

This material is based upon work supported by the Materials Computation Center at the University of Illinois funded by the NSF under Award No. 99-76550 and the Frederick Seitz Materials Research Lab funded by DOE under Award No. DEFG-96-ER45439.

6 LICENSE